Hands-on Lab: Cloning, committing and pushing your GitHub repo from the command line.

IBM Developer
SKILLS NETWORK

Effort : 30 mins

Objectives

After completing this lab you will be able to:

1. Clone your GitHub repository locally.
2. Make changes to the cloned files.
3. Add a new file.
4. Check the status.
5. Commit changes.
6. Push the changes back to GitHub.

Pre-requisites

GitHub account, with a project in it, as illustrated in the this lab.

GitBash or git installed on your local desktop, as in this lab.

Create ssh keys, as in this lab

Add SSH Key to GitHub, as in this lab

Exercise 1: Clone a repo

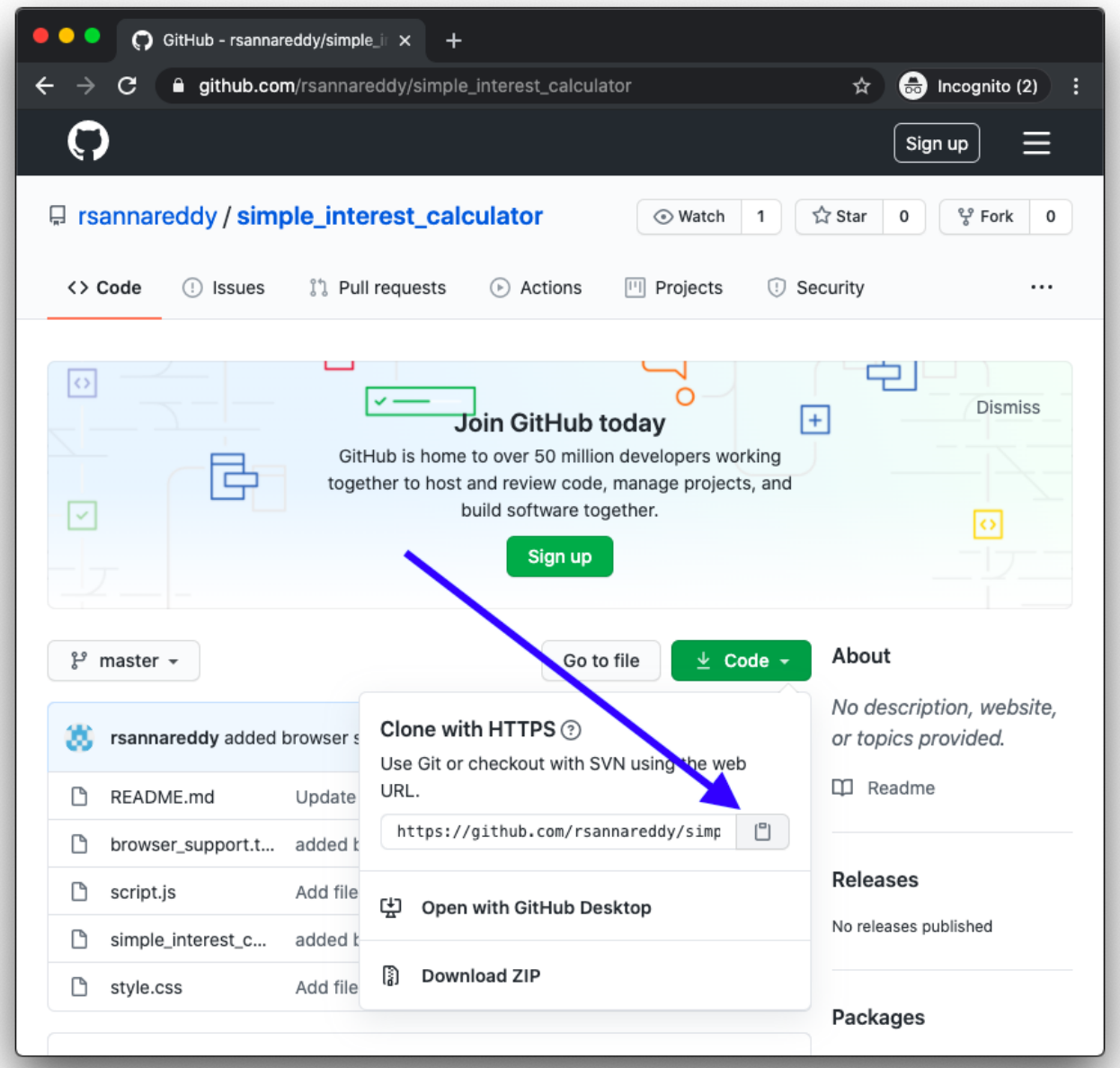
To clone a repo, you need the ssh url of the repo.

1. To get the ssh url, login into GitHub.

2. Navigate to the repo you wish to clone.

3. Click on the 'Code' button.

4. Click on the 'clipboard icon' to copy the url. Paste this url where you can access it later.



5. On your desktop open a terminal.(gitbash if you are using windows os)

6. Navigate to a directory where you wish to clone the repo.

7. Run the command "git clone "

```
work — -zsh — 99×5
(base) sr@rameshs-air work % git clone git@github.com:rsannareddy/simple_interest_calculator.git
```

8. This will clone the repo on GitHub into your current directory.

9. You can see all the downloaded files under a directory named as your repo name.

```
work — -zsh — 99×9
(base) sr@rameshs-air work % git clone git@github.com:rsannareddy/simple_interest_calculator.git
Cloning into 'simple_interest_calculator'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), done.
(base) sr@rameshs-air work %
```

change to the simple_interest_calulator directory and list the files to verify all the files got downloaded

Exercise 2: Make changes to cloned files.

Using your favourite editor make the changes to the html file.

```
simple_interest_calculator — vi simple_interest_calc.html — 99×17
<!doctype html>
<html>
    <head>
        <title>Web App to compute Simple Interest</title>
    <script src="script.js"></script>
    <link rel="stylesheet" href="style.css">
    </head>

    <body>
        <h1>Simple Interest Calculator</h1>

        <input type="number"  id="principal"> Amount <br/>
        <input type="number"  id="rate"> Rate <br/>
        <input type="number"  id="years"> No. of Years <br/>
        Interest : <span id="result"></span><br>

"simple_interest_calc.html" 21L, 542C written
```

git status will show all the modified files.

```
simple_interest_calculator — -zsh — 72×17
(base) sr@rameshs-air simple_interest_calculator % ls
README.md                       simple_interest_calc.html
script.js                       style.css
(base) sr@rameshs-air simple_interest_calculator % git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   simple_interest_calc.html

no changes added to commit (use "git add" and/or "git commit -a")
(base) sr@rameshs-air simple_interest_calculator %
```

Exercise 3: Add a new file to the local repo

Let us add a new file to the local repo.

Using a text editor, create a new file "browser-support.txt".

Add "Chrome, Firefox, Edge" into the file.

Save the file.

Exercise 4: Check the status

Run "git status" to see info on the modified files.

Let us add the file for committing.

Run "git add browser-support.txt"

Exercise 5: Commit the changes

git commit will record all the changes into the local stating area.

To commit the changes you have made. Run git commit with a message like this.

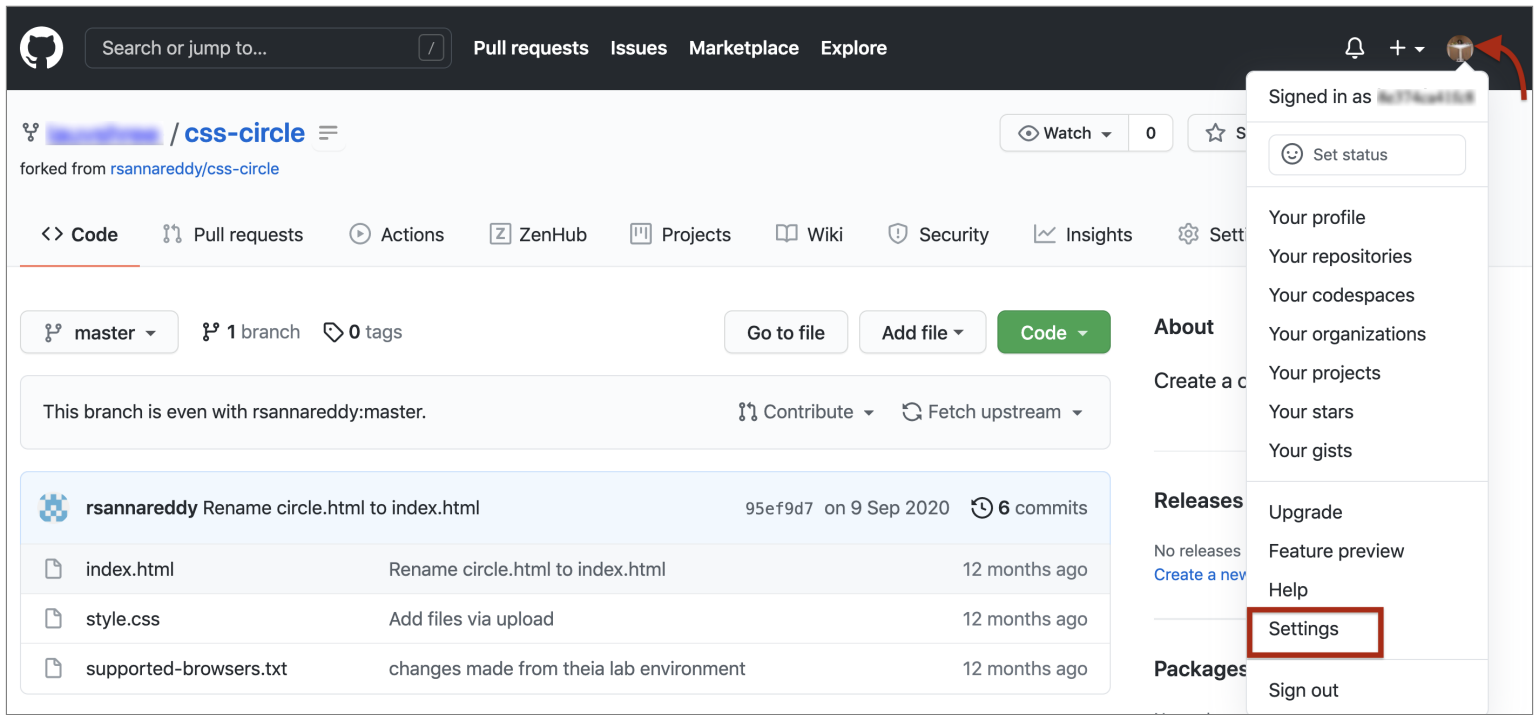
git commit -m 'added a new file browser-support.txt'

Now all the changes you have made thus far, get committed locally.

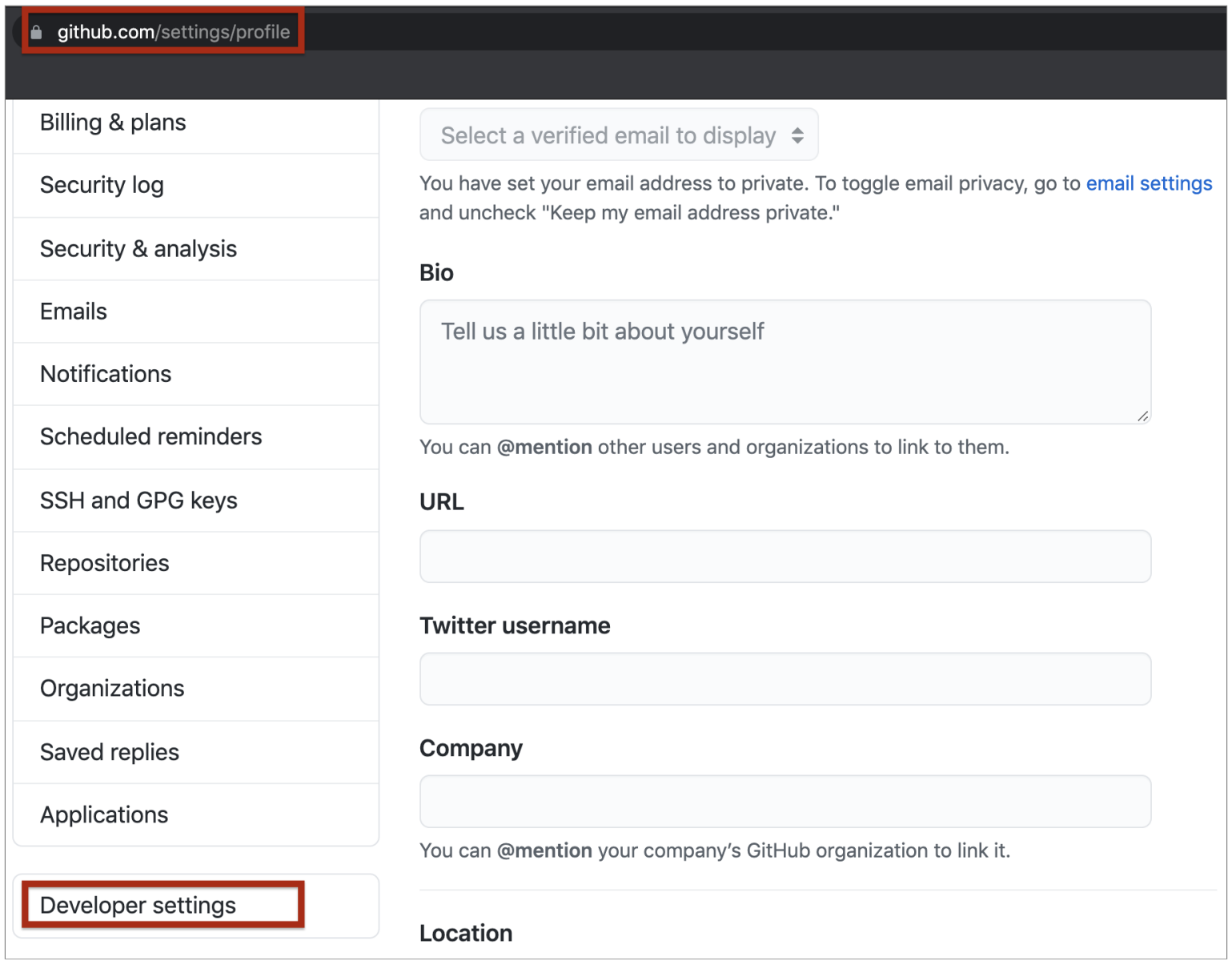
Excercise 6: Generate Personal Access Token

1. Verify your email address if it hasn't been verified on Github.

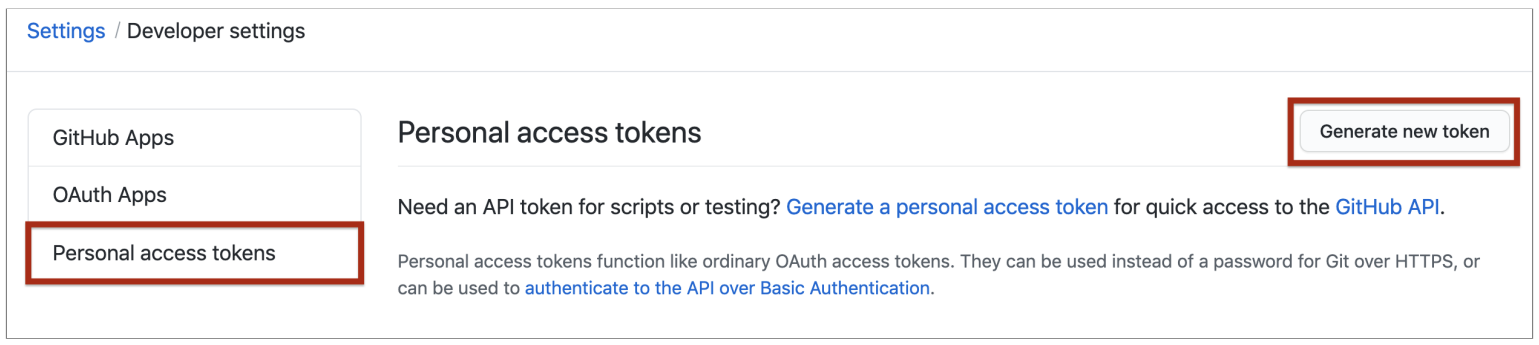
2. In the upper-right corner of any page, click your profile photo, then click Settings.



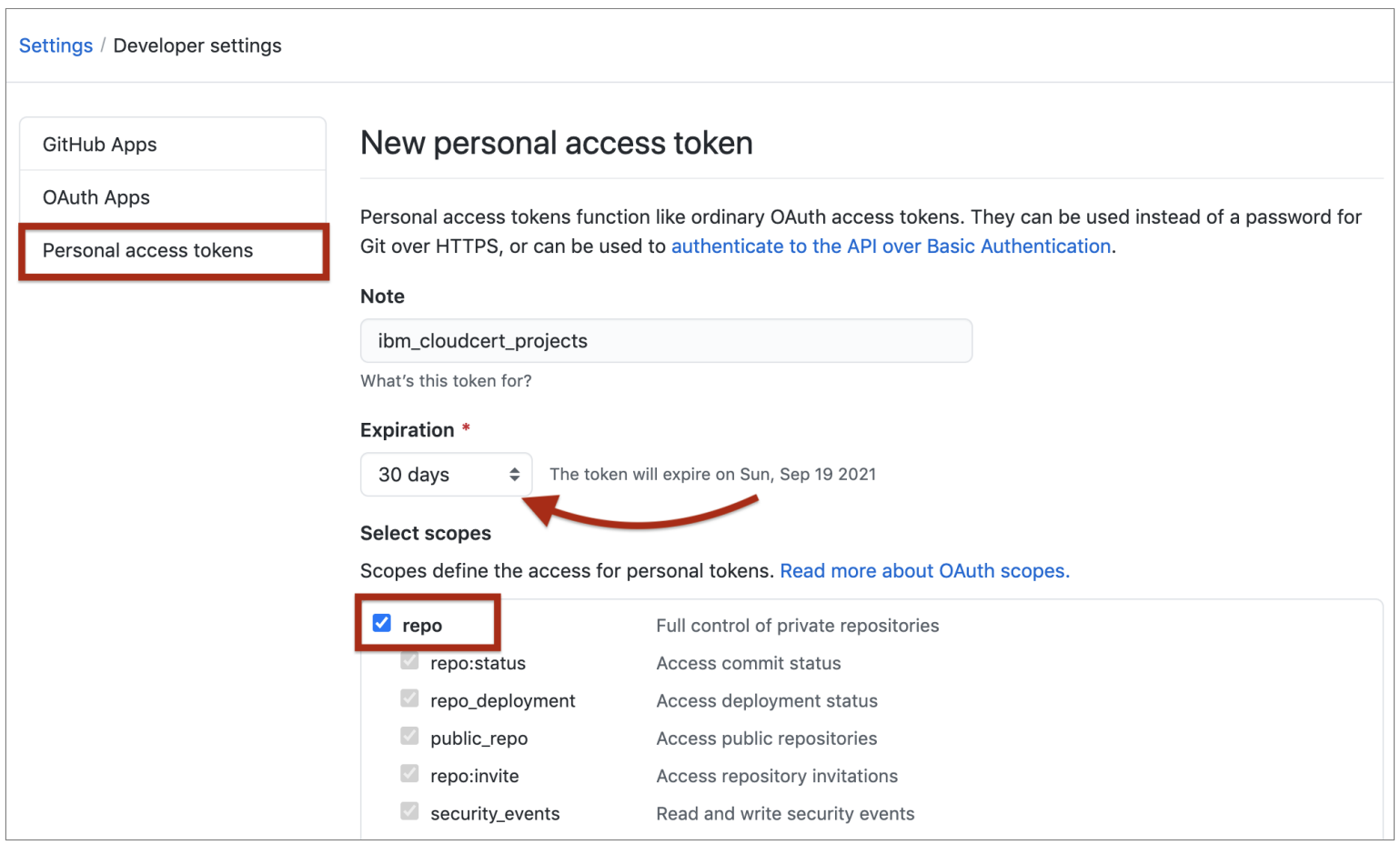
3. In the left sidebar, click Developer settings.



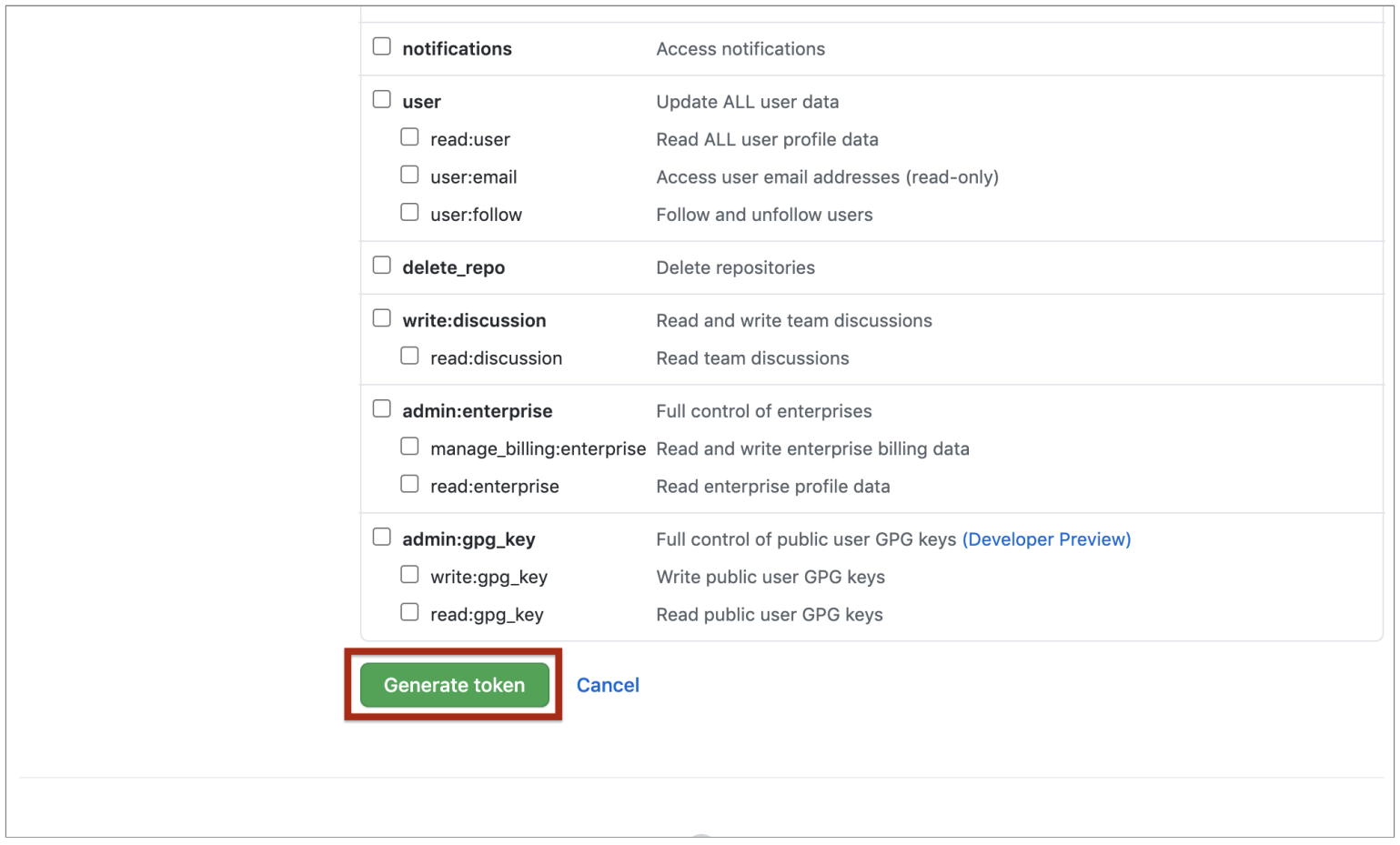
4. In the left sidebar, click Personal access tokens and click on `Generate Tokens`



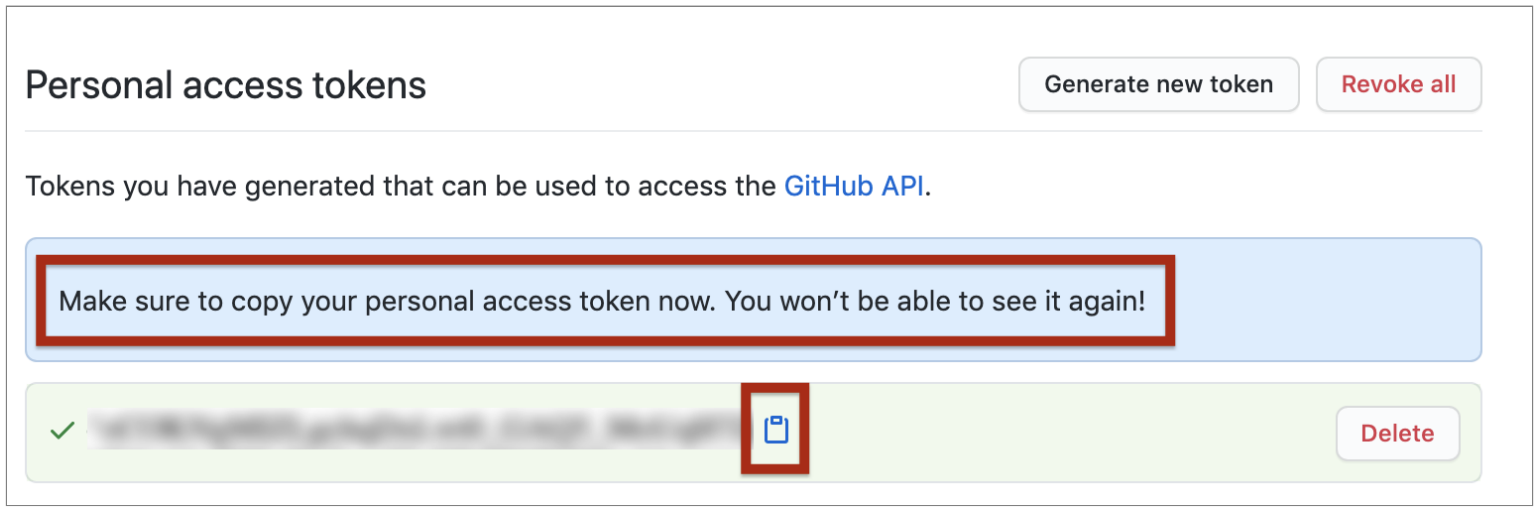
5. Give your token a descriptive name. To give your token an expiration, select the Expiration drop-down menu, then click a default or use the calendar picker. Select the scopes, or permissions, you'd like to grant this token. To use your token to access repositories from the command line, select repo.



6. Click Generate token and make a note of it.



7. Make sure you copy the token and keep it safe. It is not visible to you again.



> **Treat your tokens like passwords and keep them a secret.**

> Once you have a token, you can enter the Personal Access Token as password when performing Git operations.

Excercise 7: Push the code to GitHub

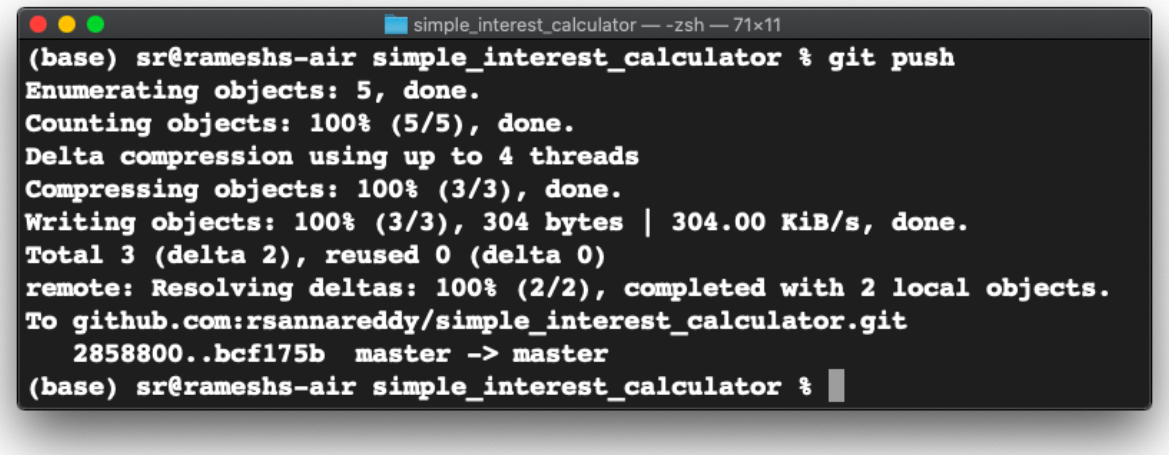
The git `push` command will enable you to sync all the changes made locally to the GitHub web repository.

```
  1. Run the following command with your actual HTTPS link:

     `git push [HTTPS link]`

     You will be prompted by git for your username and password.

  2. Type your GitHub username and for the password, enter the personal access token you generated in the previous task. When you are authenticated, all committed changes are synced
with your GitHub repository.
```



You can now visit the GitHub repository page and check to ensure that the revised and newly added files are in place.

Summary

In this lab, you have learned how to clone a GitHub repository, make changes to it, commit the changes locally, and push it back to GitHub.

Author(s)

Ramesh Sannareddy

Other Contributor(s)

Rav Ahuja

Changelog

Date	Version	Changed by	Change Description
2020-08-23	1	Ramesh Sannareddy	Initia version created.